Technical Cybersecurity

ARM & MIPS vs. x86_64

x86_64

You will never see an x86_64 CPU in IoT

- …I mean, you might, but not today
- So why are we looking at x86 and not ARM?

The concepts are the same

- …and x86 is more accessible
- You'll need to read up on the topics we've been covering for the specific processor you're targeting

ARM Differences

Register architecture

- 13 general purpose registers: *r0-r12*
- *sp*: stack pointer
- *lr*: link register, address to return to when function completes (x86 keeps this on the stack); however, in multiply nested function calls, return address for later functions is on the stack
- *pc*: program counter, points to current instruction (same as *ip* or *rip*)
- apsr: application program status register, similar to *eflags*

MIPS Differences

Register Architecture

- GP registers
  - *v0-1*: return values
  - *a0-3*: arguments to functions
  - *t0-9:* temporary registers, not preserved
  - *s0-8:* saved registers, preserved
- *gp*: global area pointer, points to global data segment
- *sp*: stack pointer
- *fp*: frame pointer
- *ra*: return address, like *lr* in arm

Instruction Architectures

ARM and MIPS are RISC

- Reduced Instruction Set Computer architectures
  - …also known as *load-store* architectures

x86 is CISC

- Complex Instruction Set Computer architecture
- However today many complex instructions are broken down into component microcode instructions

Concepts are the same, but study up on instruction and computer architectures.